LAMP-TR-067
CAR-TR-964
CS-TR-4222

MDA-9049-6C-1250
March 2001

# Relevance Ranking of Video Data using Hidden Markov Model Distances and Polygon Simplification

Daniel DeMenthon[1], Longin Jan Latecki[2], Azriel Rosenfeld[1] and Marc Vuilleumier Stückelberg[3]

[1]Center for Automation Research, University of Maryland
College Park, MD 20742-3275, USA

[2]Department of Applied Mathematics, University of Hamburg
Bundesstr. 55, 20146 Hamburg, Germany

[3]Computer Science Department, CUI, University of Geneva
24, rue Général Dufour, CH-1211 Geneva 4, Switzerland

## Abstract

A video can be mapped into a multidimensional signal in a non-Euclidean space, in a way that translates the more predictable passages of the video into linear sections of the signal. These linear sections can be filtered out by techniques similar to those used for simplifying planar curves. Different degrees of simplification can be selected. We have refined such a technique so that it can make use of probabilistic distances between statistical image models of the video frames. These models are obtained by applying hidden Markov model techniques to random walks across the images. Using our techniques, a viewer can browse a video at the level of summarization that suits his patience level. Applications include the creation of a smart fast-forward function for digital VCRs, and the automatic creation of short summaries that can be used as previews before videos are downloaded from the Web.

**Keywords:** Browsing, skimming, summarization, video segmentation, HMM, random walk, Markov model, image distance, relevance ranking, statistical representation, polygon representation, video simplification, fast-forward, video player.

This research was funded in part by the Department of Defense under Grant MDA9049-6C-1250 is gratefully acknowledged, as is the help of Sara Larson in preparing this paper.

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|---|---|---|
| **MAR 2001** | | **00-03-2001 to 00-03-2001** |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| **Relevance Ranking of Video Data using Hidden Markov Model Distances and Polygon Simplifications** | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Language and Media Processing Laboratory,Institute for Advanced Computer Studies,University of Maryland,College Park,MD,20742-3275** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES
**The original document contains color images.**

14. ABSTRACT

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT **unclassified** | b. ABSTRACT **unclassified** | c. THIS PAGE **unclassified** | | **16** | |

LAMP-TR-067
CAR-TR-964
CS-TR-4222

MDA-9049-6C-1250
March 2001

# Relevance Ranking of Video Data using Hidden Markov Model Distances and Polygon Simplification

Daniel DeMenthon, Longin Jan Latecki,
Azriel Rosenfeld and Marc Vuilleumier Stückelberg

# Relevance Ranking of Video Data using Hidden Markov Model Distances and Polygon Simplification

Daniel DeMenthon[1], Longin Jan Latecki[2], Azriel Rosenfeld[1] and Marc Vuilleumier Stückelberg[3]

[1]Center for Automation Research, University of Maryland
College Park, MD 20742-3275, USA

[2]Department of Applied Mathematics, University of Hamburg
Bundesstr. 55, 20146 Hamburg, Germany

[3]Computer Science Department, CUI, University of Geneva
24, rue Général Dufour, CH-1211 Geneva 4, Switzerland

## Abstract

A video can be mapped into a multidimensional signal in a non-Euclidean space, in a way that translates the more predictable passages of the video into linear sections of the signal. These linear sections can be filtered out by techniques similar to those used for simplifying planar curves. Different degrees of simplification can be selected. We have refined such a technique so that it can make use of probabilistic distances between statistical image models of the video frames. These models are obtained by applying hidden Markov model techniques to random walks across the images. Using our techniques, a viewer can browse a video at the level of summarization that suits his patience level. Applications include the creation of a smart fast-forward function for digital VCRs, and the automatic creation of short summaries that can be used as previews before videos are downloaded from the Web.

**Keywords:** Browsing, skimming, summarization, video segmentation, HMM, random walk, Markov model, image distance, relevance ranking, statistical representation, polygon representation, video simplification, fast-forward, video player.

---

This research was funded in part by the Department of Defense under Grant MDA9049-6C-1250 is gratefully acknowledged, as is the help of Sara Larson in preparing this paper.

## 1 Motivation

People joke that a video tape is a "Write-Only Memory" (WOM). Indeed, in many homes, hours of TV programs and family memories get videotaped and pile up, yet very little is ever viewed again. One of the reasons is that, with only fast-forward viewing as a browsing tool, it is so painfully inefficient and time-consuming to review previously recorded material or search for specific footage that it is not worth the bother. Similarly, thousands of hours of video data are becoming available online, but there is no way to quickly preview this material before committing to a complete and often lengthy download.

However, the example of text search on the Web demonstrates that even imperfect search tools can be very useful and successful. These tools attempt to rank the relevance of the search results, so that the user can focus his attention initially on material that has a higher probability of being relevant to his query.

This paper describes our approach to applying this insight to video data. We propose to summarize videos by a method that ranks frames by relevance. The proposed mechanism will let the user say "I only have the patience to download and go over the $x\%$ most useful frames of this video before I decide to download the whole video".

We would like to select frames with the highest usefulness. At first, it seems that this is hopeless, unless we can understand the semantic contents of frames and videos. For example, there might be a shot that scans over books on a shelf and stops at the title of a book that is important for understanding the story.

However, in many cases there are syntactic clues, provided by techniques that the cameraman may use to convey the importance of the shot to the story. In many cases the camera motion corresponds to the motion of the eyes of a surprised viewer. The surprised viewer's gaze is attracted to a strange part of the scene, the gaze scans the scene to "zero in" on it, zooms in on it, and dwells on it for a while, until the new information has "sunk in". These changes in the image stream can be detected without understanding the content of the stream.

In this connection, *predictability* is an important concept. Frames that are predictable are not as useful as frames that are unpredictable. We can rank predictable frames lower, since the viewer can infer them from context. Frames of a new shot cannot generally be predicted from a previous shot, so they are important. (Cuts and transitions in image streams have similarities to image edges.) On the other hand, camera translations and pans that do not reveal new objects produce frames that are predictable.

We would like to detect when the camera stops (the viewer's gaze stopping on a surprising object). Note that what is unpredictable in this case is the camera motion, not the image content. As the camera slows down, the image content stops changing, so is quite predictable. Therefore, we can consider frames in which the motion field changes as more relevant than frames in which it does not.

We turn to a signal-theoretic view of video summarization. We can assume that the original image stream signal has tens of thousands of dimensions (color components of each pixel). We apply two filtering operations. The first operation can take the form of a dimension reduction that finds a feature vector for each frame and transforms the image stream into a feature vector trajectory, a signal that has many fewer dimensions than the

original signal (e.g., 37 in one of the methods described below). Alternatively, we can represent each frame by a statistical model that captures average characteristics of colors, and possibly texture and motion, as well as contiguity properties. In this method too, we can view the original image stream as being filtered into a new signal in some (non-Euclidean) space, where we define the distance between frames as the distance between their statistical models. Both of these methods are described in the next section.

As a result of the first filtering step, we would like the output signal to be a straight line, or to remain in approximately the same position in the space, when nothing of interest happens, and to have a detectable curvature, a step or a roof, when a noteworthy event takes place.

Noise in this context is not the same as pixel noise. The image stream generated by a fixed camera looking from a window at a crowd milling around in the street may be considered to have a stationary component and a visual noise component, due to the changing colors of people's clothes. The passing of a fire truck would be an example of a signal over this fluctuating but monotonous background.

We apply a second filtering step, with the goal of detecting regions of high curvature along the trajectory, and we rank the filtering results. Since we expect the video signal to be noisy in the sense described above, we need the second filtering step to enhance the linear parts as well as the parts with significant curvature. In the non-Euclidean feature space of statistical frame models, projections cannot be computed; thus this filtering step should use only distance measures between models. It should allow for hierarchical output, so that the user can specify the level of detail (or scale) at which he wants to view the frames that show noteworthy events.

We can attempt to optimize both the first filtering step (mapping to the feature vector trajectory) and the second filtering step (edge-roof detection).

## 2 Mapping an Image Stream into a Trajectory

We begin with our proposal for the first filtering step, motivated in the previous section. We present two mappings of an image stream into a trajectory such that the trajectory is highly bent when events of interest occur in the stream. We assign a point on the trajectory to each frame in the stream.

For the first mapping, we define four histogram buckets of equal size for each of the three color attributes in the YUV color space of MPEG encoding. Each bucket contributes three feature vector components: the pixel count, and the $x$ and $y$ coordinates of the centroid of the pixels in the bucket. This yields 36 components, and we add the frame number (time) to obtain 37 components. Thus, the trajectory in this case is a polygonal arc in $\mathbb{R}^{37}$. (We are investigating an alternate scheme in which the number and sizes of the buckets are selected according to the color distribution over the video sequence.)

When the camera translates or pans smoothly without seeing new things, the centroid components change linearly and the feature vector trajectory is linear. If the camera suddenly decelerates (or accelerates), the trajectory has a high curvature, because the centroids decelerate.

As an alternative mapping, we generate a statistical model for each frame using a hidden Markov model (HMM) technique. We obtain sequences of pixels by taking random walks across each frame, moving either to the north, east, south or west neighbor with equal probability. When we hit the border of the image, we jump randomly to another pixel of the image, and start another sequence. We model our observations by counting the number of times we step to a pixel of a certain color, given that we come from a neighbor of the same color or of a different color (colors are the same if they are quantized to the same value; the quantization method is described below). Numbers representing the counts of transitions from one color to another color can be stored in a 2D table. Note that this table is a cooccurrence matrix [6] for the quantized colors, except for the fact that some pixels my be visited twice and other pixels may be missed.

To make this information independent of the number of steps taken, we can normalize each row so that the row numbers sum to 1. For large numbers of observations, this table is a color transition probability matrix, as it describes the probability of arriving at a certain color at the next step, given that we are at a certain color at the present step. In addition, we keep track of the values of the pixels at the first step of each new walk to compute a histogram of the colors of the image.

To avoid excessive model size, the colors must be quantized. Using HMM terminology, this operation can be called a *state assignment* of the pixels, since we are saying that when a color is in a certain interval, the pixel belongs to a given bin, or state. After quantization, we can describe the image by a histogram of the states and a state transition matrix. To compensate for the reduced descriptive power of a statistical model using fewer states, the HMM describes the distribution of each color within each bin/state. In our experiments, we modeled the color distribution within each state by three Gaussians, i.e. a total of six numbers. HMM techniques (described in the next paragraph) allow us to compute a quantization of the color space such that in each bin/state, the color distributions are well represented by Gaussians. The labeling of pixels into states is hidden, in the sense that only actual pixel values are observed, not their quantized values, and a computation assigns the best states as follows.

A state assignment is obtained in two steps in an iteration loop. In the first step, we compute the sequences of states that have the highest probabilities, given the observation sequences along the random walks. We obtain these maximum probabilities and the corresponding state sequences by a dynamic programming technique called the Viterbi algorithm [12], using the state transition matrix and the probability distribution of observations within each state (obtained at a previous iteration). In the second step, now that each pixel has been labeled with a specific state, we can recompute the most likely state transition matrix by tallying the transitions from state to state along the random walks. Also, we can recompute the most likely Gaussian probability distributions of observations within each state by finding the means and variances of the colors of the pixels labeled with that state. These two steps are repeated alternatingly until there is no significant improvement. This is the so-called segmental K-means approach to computing a Hidden Markov model from sequences of observations [12]. (The slower-converging Baum-Welch algorithm can be used instead with similar results.) The resulting statistical description of the image consists of a state transition matrix, which is essentially a cooccurrence matrix for quantized colors, together with a description of the color distributions within

each bin, and the probability distribution of the states of the starting pixels of each random walk, which is a histogram of the quantized colors of the image.

Once we have obtained HMM models for the video frames, we are able to compute distances between frames. The idea behind a distance calculation between two images using HMMs is to find how well the HMM of one image can model the other (and vice versa), in comparison with how well each HMM can model the image on which it was trained. To measure the modeling ability of an HMM for any image, we can obtain an observation sequence from that image by a random walk, and compute the probability that this sequence could be produced by the HMM. When images are visually similar, this probability is high.

In other words, a distance measure between two images $I_1$ and $I_2$ with HMM models $\lambda_1$ and $\lambda_2$ is constructed by combining the probability that the observation sequences $\mathbf{O}_2$ obtained by random walks through image $I_2$ could be produced by the probabilistic model $\lambda_1$ of image $I_1$, and a similar probability where the roles of $I_2$ and $I_1$ are reversed. This quantity is normalized by the number of observations, and compared to how well the HMMs can model the images on which they were trained:

$$d(I_1, I_2) = -\frac{1}{N_2}\log P(\mathbf{O}_2|\lambda_1) - \frac{1}{N_1}\log P(\mathbf{O}_1|\lambda_2) + \frac{1}{N_1}\log P(O_1|\lambda_1) + \frac{1}{N_2}\log P(O_2|\lambda_2) \tag{1}$$

Quantities of the form $\log P(\mathbf{O}_i|\lambda_j)$ are computed by applying the classic Forward Algorithm [12] to the observation sequences $\mathbf{O}_i$ using the transition matrix and probability distributions prescribed by the HMM model $\lambda_j$.

This distance function defines a *semi-metric space*, because it satisfies
positivity: $d(x,x) = 0$ and $d(x,y) > 0$ if $x$ is distinct from $y$,
symmetry $d(x,y) = d(y,x)$,
but not the triangle inequality, i.e., there can exist $z$'s such that

$$d(x,y) > d(x,z) + d(z,y).$$

For this mapping, the trajectory describing a sequence of video frames is also a polygonal arc (in the sense that it is a finite, linearly ordered sequence of points) but it is not contained in Euclidean space; it is contained in a non-linear semi-metric space. This means that the points on the trajectory cannot be assigned coordinates, and we can only measure a semi-distance between any two points.

Distances based on image statistics (histogram, co-occurrence, HMM) are quite insensitive to image translation, and therefore produce points that are in approximately the same positions in space when the camera motion is a pan or a translation.

## 3 Trajectory Filtering by Polygon Simplification

Our first filtering operation, described in the previous section, maps a video sequence into a trajectory that is a polygonal arc, i.e., a polyline. The polyline may be noisy, in the sense that it is not linear but only nearly linear for video stream segments where nothing of interest happens (i.e., where the segments are predictable). Furthermore, the parts of high curvature are difficult to detect locally. Therefore, it is necessary to apply a second filtering operation, which we describe in this section.

The goal is to simplify the polyline so that its sections become linear when the corresponding video stream segments are predictable, which also means that the vertices of the simplified polyline are key frames of the non-predictable video stream segments. We achieve this by repeated removal of the vertices that represent the most predictable video frames. In terms of the geometry of the polyline trajectory, these vertices are the most linear ones. While it is clear what "linear" means in a linear space, we need to define this concept for semi-metric non-linear spaces.

A polyline is an ordered sequence of points. Observe that even if the polyline is contained in Euclidean space, it is not possible to use standard approximation techniques like least-square fitting for its simplification, since the simplified polyline would then contain vertices that do not belong to the input polyline. For such vertices, there would not exist corresponding video frames. Thus, a necessary condition for a simplification of a video polyline is that the sequence of vertices of the simplified polyline be a subsequence of the original one.

Our approach to simplification of video polylines is based on a novel process of discrete curve evolution presented in [9] and applied in the context of shape similarity of planar objects in [11]. However, here we will use a different measure of the relevance of vertices, described below.

Aside from its simplicity, the process of discrete curve evolution differs from the standard methods of polygonal approximation, like least square fitting, by the fact that it can be used in non-linear spaces. The only requirement for discrete curve evolution is that every pair of points is assigned a real-valued distance measure that does not even need to satisfy the triangle inequality. Clearly, this requirement is satisfied by our distance measure, which is a dissimilarity measure between images.

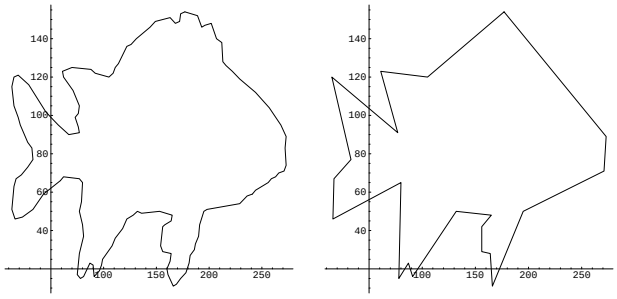
Now we briefly describe the process of discrete curve evolution (for more details see [10]). The basic idea of the proposed evolution of polygons is very simple:

- At each evolution step, the vertex with smallest relevance is detected and deleted.

The key property of this evolution process is the order of the deletion, which is given by a relevance measure $K$ that is computed for every vertex $v$ and depends on $v$ and its two neighbor vertices $u, w$:

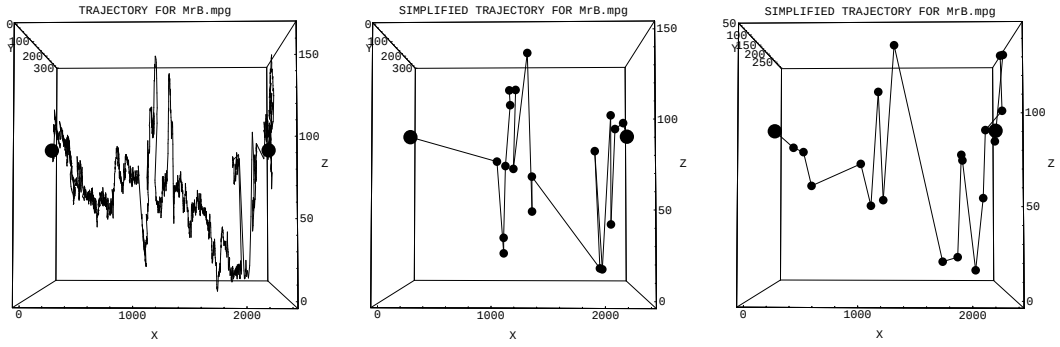
$$K(v) = K(u, v, w) = d(u, v) + d(v, w) - d(u, w) \tag{2}$$

where $d$ is the semi-distance function. Intuitively, the relevance $K(v)$ reflects the shape contribution of vertex $v$ to the polyline.



(a) (b)

Figure 1: Fish silhouette with 124 vertices (a) and a simplified curve with 21 points (b)




(a) (b) (c)

Figure 2: Video trajectory (a) and curve simplification producing 20 relevant frames (black dots) for "Mr. Bean's Christmas", using (b) the feature vector method, and (c) the HMM method.

Fig. 1 illustrates the curve simplification produced by the proposed filtering technique for a planar figure. Notice that the most relevant vertices of the curve and the general shape of the figure are preserved even after most of the vertices have been removed.

We will demonstrate with the experimental results in the next section that the discrete curve evolution based on this relevance measure is very suitable for filtering polylines representing videos.

## 4 Experimental Results

We illustrate our techniques using an 80-second clip from a video entitled "Mr. Bean's Christmas". The clip contains 2379 frames. First, we applied the feature vector approach described above, in which a 37-dimensional feature vector derived from centroids and pixel counts in histogram bins is computed for each frame. A perspective view of the 3D projection of the video trajectory is shown in Fig. 2a. The two large black dots are the points corresponding to the first and last frames of the video. Curve simplification using the method described in Section 3 was then applied to this trajectory. Fig. 2b shows a simplification result in which only 20 points have been preserved. A method for

automatic selection of the smallest point count that can still provide an appropriate level of summarization is presented at the end of the next section.

Finally, HMM models were computed for every five frames, and curve simplification was performed using the probabilistic distance measure described in Section 2. For comparison with the feature vector method, we chose to also preserve 20 key frames with the HMM curve simplification. Since this method does not provide frame coordinates, we plotted the 20 points that correspond to these 20 frames along the trajectory found by the feature vector approach, in order to give an idea of the locations of the 20 frames in the video (Fig. 2c). Clearly, the HMM method located its key frames on segments of sudden change of that trajectory, i.e. in regions of significant change in the video clip.

Next we discuss the quality of the summaries produced by curve simplification using the feature vector and HMM methods. Knowing the content of each shot of the clip is helpful for this comparative evaluation.

1. Frames 1 to 996: Mr. Bean carries a raw turkey from a kitchen counter to a table. He cuts a string that tied the legs, brings a bowl of stuffing closer and starts pushing stuffing inside the turkey.

2. Frames 997 to 1165: He notices that his watch is missing.

3. Frames 1166 to 1290: He looks inside the turkey, then pulls stuffing out of the turkey to retrieve his watch.

4. Frames 1291 to 1356: He keeps removing stuffing.

5. Frames 1357 to 2008: He tries to look inside, then uses a flashlight to try to locate his watch inside the turkey. Finally, he bends toward the turkey to explore more deeply by putting his head inside the turkey.

6. Frames 2009 to 2079: The lady friend whom he invited for Christmas dinner rings his doorbell.

7. Frames 2080 to 2182: Hearing the bell, Mr. Beans stands up with his head stuck inside the turkey. He vainly attempts to remove the turkey.

8. Frames 2183 to 2363: He walks blindly toward the door with the turkey over his head, bumping into things.

9. Frames 2364 to 2379: His lady friend waits outside for the door to open...

Fig. 3 shows two storyboards obtained by curve simplification. Storyboard (a) results from the curve simplification obtained by the feature vector method. The frames correspond to the vertices of the simplified polyline in Fig. 2b. Storyboard (b) results from the HMM method. The frames correspond to the vertices of the simplified polyline in Fig. 2c.

Both storyboards seem to be reasonable summarizations of the short video clip. The feature vector method misses Shot 9 (the last shot), and oversamples Shot 2 (where he notices that his watch is missing) with 8 frames. The cause of this oversampling was

traced to the histogram quantization of the feature vector method (Section 2). Pixels of colors located half-way between the representative colors of histogram bins could switch their assignments from one bin to the next with a small variation of color between successive frames. In rare instances, a relatively large number of pixels may flip back and forth between two neighboring bins, causing large jumps in the feature vector components which result in large meaningless peaks in the polygonal line representing the video. The HMM method is not affected by such quantization artifacts; it represents each shot with at least one frame, and selects frames that tell more of the story, such as the string cutting of frame 381 and the flashlight episode of frame 1771. A more thorough evaluation could be obtained by comparison with ground-truth provided by humans who view the clips and select small numbers of frames as most descriptive of the stories.

## 5 A Video Player with Smart Fast-Forwarding

An interesting application of video summarization is to the design of a smart VCR fast-forwarding that samples only the most relevant frames. We have developed a Java video player that plays video clips in MPEG format, and can play the whole video at the normal rate, or show only the frames of highest relevance(Fig. 4a). A vertical *relevance slider* on the right-hand side of the window lets the user define the number of frames that he has the patience to watch. For example, the "Mr. Bean's Christmas" video clip contains 2379 frames, so that playing it takes around 80 seconds. The user may choose to watch only the 20 most relevant frames and moves the slider up until the box at the left of the sliding elevator indicates 20. Then the player skips all but the 20 most relevant frames. The buttons at the bottom of the player window define VCR-type functions: Play, Pause, Fast-Forward, Fast-Backward, frame-by-frame forward stepping, and backward stepping. In all these modes, only the relevant frames, as defined by the relevance slider, are played.

A horizontal *sampling stripe* located under the image display panel shows the positions of the relevant frames within the video. It is a black stripe that shows a white vertical tick mark for each displayed frame. A triangular *frame marker* slides below the sampling stripe as the video clip is being played, and indicates which frame is being displayed. Navigation through the video can also be performed by dragging this triangular frame marker. This mode of navigation is called "scrubbing" by video editing practitioners. It is set to let the user visit all the frames, not just the relevant frames.

Video clips are selected from a pop-up menu. The user can also select different types of relevance measures from a second pop-up menu. The relevances presently available in the video player have been precomputed from Euclidean distances between feature vectors from histogram bins and from HMM distances, both described above, as well as from Euclidean distances between motion feature vectors, described in [16]. We plan to add other filtering choices, such as relevances based on the presence of faces, music and talk content in the sound track.

When a new video is selected for viewing, the vertical relevance slider is initially positioned at a default position which shows only a small number of relevant frames. This number is precomputed for each available type of relevance measure using a histogram slope technique. Cumulative histograms that represent, for any given relevance, the number of frames that have larger relevance, are found to have similar shapes (Fig. 4 (b)):

most of the frames have small relevances; these frames have small variations with respect to their neighbors. Very few frames have large relevances. The two regions are separated by a sudden slope change. We wish to ignore the many frames with small relevances and show the few frames with large relevances; therefore we select the cutoff relevance at the slope break between the two regions, around slope $-1$. For the histogram of Mr. Bean's video, this corresponds to around 27 frames. After exploring the video at several relevance slider positions, the user can return the slider to its default position by clicking the button labeled "Reset Sampling".

This prototype uses MPEG-decoding Java source code written by J. Anders [1].

## 6 Related Work and Discussion

Huang et al. [8] showed that using more descriptive statistical models of images such as correlograms significantly improves retrieval performance of images, in comparison to simple statistical descriptions such as histograms. However, they do not have a method for selecting the right balance between the size of the correlogram and the discriminative power of the model. Second, they apply Euclidean distances to their statistical models. Consequently, they have to give different weights to the Euclidean coordinates depending on the situation. In our view, the use of hidden Markov models of images elegantly addresses these issues, by (1) supplementing coarse color quantization with a description of color distributions within each bin, while automatically adjusting each bin size to make this description optimal, and (2) addressing the distance issue by allowing an intuitive probabilistic definition of image distance.

In [3], we described how the Ramer-Douglas-Peucker method of polygon simplification [13] could provide effective summarizations of videos. This method is a binary curve splitting approach that at each step splits the arc at the point furthest from the chord, and stops when the arc is close to the chord. However, for $N$ video frames it has time complexity $N^2$, which is prohibitive for large videos and complex distance measures. Variants that reduce the complexity to $N \log N$ cannot be applied to multidimensional video trajectories, as they make use of planar convex hulls [7]. In addition, the computation of the distance between an arc and its chord requires the use of Euclidean distances. The curve simplification technique we have proposed can be shown to be of order $N \log N$ and can accommodate non-Euclidean distances. These two features make the use of probabilistic image distances practical for video summarization.

The reader interested in video browsing research can refer to [14, 15, 17] and to the recent work of Foote [5].

## 7 Conclusions and Future Work

In this work, we have proposed and implemented a system for automatically providing summaries of videos whose size can be controlled by the user. The method applies a novel fine-to-coarse polyline simplification technique that computes for each vertex a relevance measure based on its two neighbors and at each step removes the least relevant vertex and updates the relevances of the affected neighbors. The proposed relevance

measure is valid for non-metric spaces. This allows us to compute relevances using a probabilistic distance measure between hidden Markov models of the video frames. We produce reasonable summaries by showing the most relevant frames in temporal order. We have implemented a video player that incorporates this technology to let the user perform a smart fast-forwarding that skips the more predictable frames. A vertical slider lets the user define the number of relevant frames he has the patience to watch. We are currently investigating improved random sampling of images for the HMM calculation using quasi-random walks [2], as well as summarization results for a 2D HMM technique [4]. We are also improving our video player to let the user select a region of a frame and retrieve the frames that have the shortest HMM distances to that region.

## References


[1] Anders, J., Java MPEG page, http://rnvs.informatik.tu-chemnitz.de/~ja/MPEG/MPEG_Play.html

[2] Coulibaly, I., and Lécot, C.,"Simulation of Diffusion Using Quasi-Random Walk Methods", Mathematics and Computers in Simulation, vol. 47, pp. 154–163, 1998.

[3] DeMenthon, D.F., Kobla, V., M., and Doermann, D., "Video Summarization by Curve Simplification", Technical Report LAMP-TR-018, CAR-TR-889, July 1998; also ACM Multimedia 98, Bristol, England, pp. 211-218, 1998.

[4] DeMenthon, D.F., Vuilleumier Stückelberg, M., and Doermann, D., "Hidden Markov Models for Images", Int. Conf. on Pattern Recognition, Barcelona, Spain, 2000.

[5] Foote, J., Boreczky, J., Girgensohn, A., and Wilcox, L. (1998), "An Intelligent Media Browser using Automatic Multimodal Analysis", ACM Multimedia 98, Bristol, England, pp. 375-380, 1998.

[6] Haralick, R.M., "Statistical and Structural Approaches to Texture", *Proceedings of the IEEE*, vol. 67, pp. 786–804, 1979.

[7] Hershberger, J., and Snoeyink, J. "Speeding up the Douglas-Peucker Line-Simplification Algorithm", http://www.cs.ubc.ca/cgi-bin/tr/1992/TR-92-07.

[8] Huang, J., Kumar, S.R., Mitra, M., Zhu, W-J., and Zabih, R., "Image Indexing Using Color Correlograms", IEEE Conf. on Computer Vision and Pattern Recognition, pp. 762–768, 1997.

[9] Latecki, L.J., and Lakämper, R., "Convexity Rule for Shape Decomposition based on Discrete Contour Evolution", Computer Vision and Image Understanding, vol. 73, pp. 441–454, 1999.

[10] Latecki, L.J., and Lakämper, R., "Polygon Evolution by Vertex Deletion", in M. Nielsen, P. Johansen, O.F. Olsen, and J. Weickert, editors, Scale-Space Theories in Computer Vision (Int. Conf. on Scale-Space), LNCS 1682, Springer, 1999.

[11] Latecki, L.J. and Lakämper, R., "Shape Similarity Measure Based on Correspondence of Visual Parts", IEEE Trans. on Pattern Analysis and Machine Intelligence, to appear.

[12] Rabiner, L.R., and Juang, B.-H., "Fundamentals of Speech Processing", Prentice Hall, pp. 321–389, 1993.

[13] Ramer, U., "An Iterative Procedure for the Polygonal Approximation of Plane Curves", Computer Graphics and Image Processing, vol. 1, pp. 244–256, 1972.

[14] Smith, M.A., and Kanade, T., "Video Skimming for Quick Browsing Based on Audio and Image Characterization", IEEE Conf. on Computer Vision and Pattern Recognition, 1997.

[15] Yeung, M.M., Yeo, B-L., Wolf, W. and Liu, B.,"Video Browsing using Clustering and Scene Transitions on Compressed Sequences", SPIE Conf. on Multimedia Computing and Networking, vol. 2417, pp. 399–413, 1995.

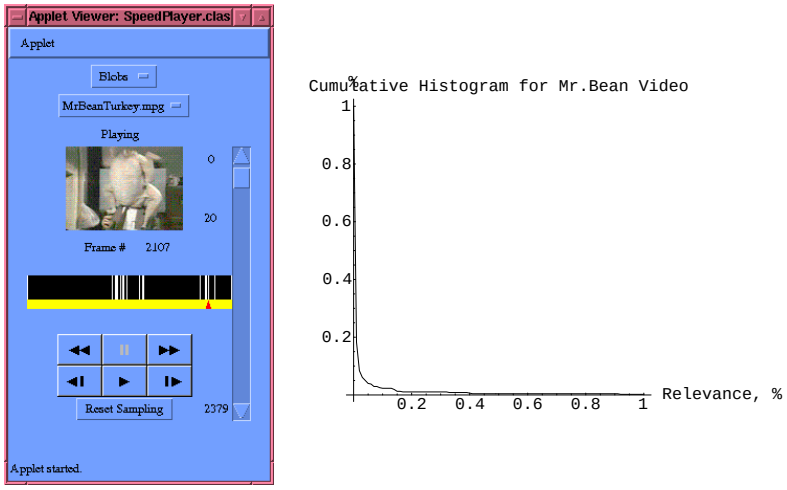
[16] Yoon, K., DeMenthon, D.F., and Doermann, D., "Event Detection from MPEG Video in the Compressed Domain", Int. Conf. on Pattern Recognition, Barcelona, Spain, 2000.

[17] Zhang, H.J., Low, C.Y., Smoliar, S.W., and Wu, J.H., "Video Parsing, Retrieval and Browsing: An Integrated and Content–Based Solution", ACM Multimedia, 1995.


(a)


(b)

Figure 3: Storyboards obtained by curve simplification of the video trajectory obtained by the feature vector method (a) and by the HMM method (b)




(a) (b)

Figure 4: (a): Video player with vertical slider for control of summarization level. (b): Cumulative histogram giving proportion of frames with relevances larger than a given number, used to determine default summarization level.